Sarcouncil Journal of Engineering and Computer Sciences

ISSN(Online): 2945-3585

Volume- 04| Issue- 08| 2025

Research Article **Received:** 25-06-2025| **Accepted:** 16-07-2025 | **Published:** 04-08-2025

Demystifying Software-Defined Vehicle Architecture for Modern Automotive Systems

Narendra Babu Atmakuri
Independent Researcher, USA

Abstract: The article "Demystifying Software-Defined Vehicle Architecture for Modern Automotive Systems" examines the transformative evolution in automotive design as vehicles transition from hardware-centric platforms to software-defined systems. It explores how the Software-Defined Vehicle (SDV) paradigm is revolutionizing automotive architecture through centralized computing platforms, virtualization technologies, and service-oriented design principles. The article details the layered software stack that enables hardware abstraction, the critical role of hypervisors in managing mixed-criticality workloads, and the heterogeneous computing approaches that optimize performance across diverse vehicle functions. The article investigates enabling technologies, including over-the-air update infrastructure and standardized middleware frameworks that facilitate continuous improvement throughout vehicle lifecycles. It extends to real-world applications, highlighting accelerated innovation cycles, hardware consolidation benefits, and emerging business models enabled by this architectural approach. The article also addresses key challenges in functional safety compliance, cybersecurity, and development complexity while providing perspective on future directions, including edge-cloud hybrid computing and specialized automotive processing technologies.

Keywords: Software-Defined Vehicles, Automotive Architecture, Over-the-Air Updates, Functional Safety, Heterogeneous Computing.

INTRODUCTION

Cars have changed dramatically over the last decade. Gone are the days when mechanical prowess defined automotive excellence. Today's vehicles bear more resemblance to rolling computers than traditional automobiles. This revolution has birthed the Software-Defined Vehicle (SDV)—a radical break from conventional design where digital capabilities, not mechanical specifications, determine what a car can do.

Look inside any luxury vehicle today, and you'll find digital systems controlling everything from fuel injection to cabin temperature. This explosion of digital complexity has pushed traditional control architectures to their breaking point. Manufacturers find themselves abandoning distributed systems for consolidated computing frameworks that can handle increasingly sophisticated software demands while changing competitive dynamics across the industry (Rinf Tech,).

This digital transformation reshapes automotive economics at its core. As software becomes the heart of vehicle identity, development cycles shrink from years to months or even weeks. Forward-thinking automakers have abandoned rigid annual update cycles, instead rolling out features continuously throughout a vehicle's life. This flexibility creates fresh revenue streams through digital feature activation and subscription models that simply weren't possible when functions were permanently locked into hardware (RinfTech,).

These new generation makes are integrating functions into potent domain controllers that are structured around key car systems in the hood. Combining conventional processors and specialized chips performing operations such as artificial intelligence, signal processing, and graphics rendering, these computing hubs are optimized to meet the needs of the many distinct applications in various domains. This heterogeneous computing effectively supports a wide range of operational requirements but isolates the safety-critical functionality of the vehicle to allow advanced driver assistance as well as high levels of connectivity (Charette, R. N. 2009).

For development teams, this paradigm demands profound methodological shifts. Waterfall approaches yield to agile frameworks supporting rapid innovation. This transition requires reinvented testing approaches, ensuring absolute safety while accommodating faster release schedules. Modern automotive software development involves sophisticated simulation environments, testing changes across countless scenarios before actual vehicles see a single line of new code (Charette, R. N. 2009).

As cars evolve into smart, connected platforms, understanding SDV architecture becomes essential for anyone involved in transportation's future. The revolution is accelerating, as cars grow to be more and more driven by software, a key distinguishing factor when it comes to competing cars and

***Corresponding Author:** Narendra Babu Atmakuri

DOI- https://doi.org/10.5281/zenodo.16735475

automobiles-the real computers on wheels, designed with the next generation roads in mind.

THE PARADIGM SHIFT: FROM HARDWARE-CENTRIC TO SOFTWARE-DEFINED

Traditional cars had to run their functions using individual electronic control units (ECUs) that did not communicate in a robust manner. Modern SDVs take a radically different approach, implementing unified computing architectures with powerful domain controllers that integrate functions previously scattered across dozens of separate modules. This consolidation creates platforms where software masks hardware complexity while maintaining stringent safety requirements.

This architectural revolution breaks decisively from decades of automotive design tradition. Conventional vehicle frameworks featured single-purpose ECUs running proprietary code with limited cross-system communication. Premium vehicles ended up housing over 150 separate ECUs running more than 100 million lines of code from various suppliers. This fragmented approach reached practical sustainability limits as vehicles demanded increasingly sophisticated capabilities. The transition toward function-organized domain controllers allows manufacturers to implement streamlined architectures with standardized interfaces. Industry research suggests this consolidation cuts development costs nearly in half while dramatically speeding innovation. The software-defined approach creates solid foundations for advanced driver assistance features and eventual autonomous driving capabilities, requiring sophisticated cross-domain integration (Burkacky, O. *et al*., 2018).

The software-centric paradigm brings remarkable advantages in development flexibility and operational efficiency. By creating standardized software interfaces that hide hardware details, manufacturers can add features and enhance capabilities throughout a vehicle's life. This capacity enables deployment of performance improvements and problem fixes without physical modifications. Transitioning to software-defined approaches requires fundamental shifts in development organization, embracing continuous integration methods borrowed from tech companies. This evolution creates opportunities for novel business models, including on-demand feature activation and subscription services impossible under traditional frameworks. The software-defined approach lets manufacturers respond quickly to market shifts and consumer preferences while establishing ongoing revenue streams beyond initial purchase (Kochhar, N. 2024).

Automotive experts describe this transition as the most significant architectural transformation since electronics first appeared in vehicles. The software-defined approach delivers unprecedented flexibility in implementing and updating vehicle functions throughout product lifespans.

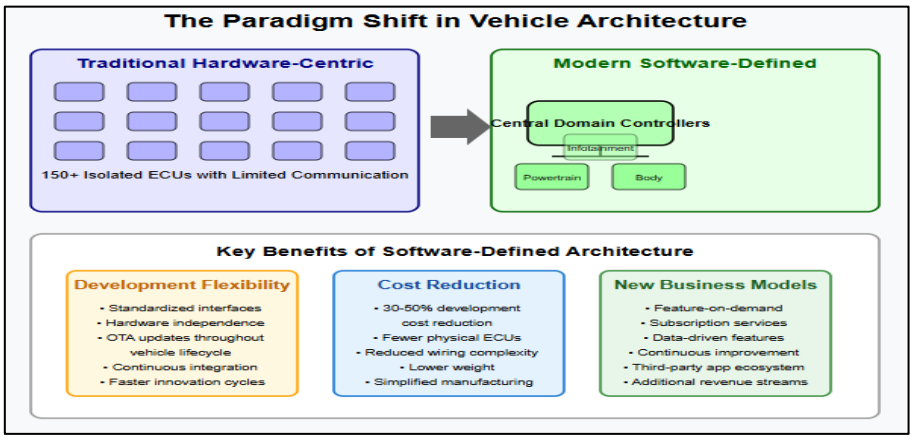

Fig 1: The Paradigm Shift: From Hardware-Centric to Software-Defined Vehicles (Burkacky, O. *et al*., 2018; Kochhar, N. 2024)

CORE COMPONENTS OF SOFTWARE-DEFINED VEHICLE ARCHITECTURE

Layered Software Stack

SDV architecture comprises several distinct layers working together to create adaptable, updatable platforms. This tiered approach lets different components evolve independently while preserving system integrity. The automotive stack of the software reuses the ideas of enterprise

Publisher: SARC Publisher

computing, but deals with the specific automotive needs.

An underlying Hardware Abstraction Layer (HAL) standardises the interfaces to vehicle sensors, actuators, and computing resources and serves to abstract the software from the physical hardware. This abstraction guarantees portability of software, giving manufacturers the ability to update or replace components without rewriting the code. Over this, the Operating System Layer provides execution environments appropriate to various vehicle functions, with real-time operating systems being typically utilized when a performance-predictability is necessary, such as with safety-critical functions (e.g., braking), and general-purpose operating systems being used when more-generic functionality is required, such as infotainment. Integrating these environments requires sophisticated partitioning mechanisms ensuring entertainment applications cannot interfere with safety-critical functions while enabling efficient resource sharing (Mobility, B.).

The Middleware Layer functions as the communication backbone connecting different software components, offering standardized mechanisms for data exchange, service discovery, and resource management. This tier implements automotive-specific protocols enabling interoperability between components from different suppliers. At the highest level, the Application Layer houses actual vehicle functions, including driver assistance features, powertrain control algorithms, and user interfaces. This modular organization allows specific functions to receive updates without disrupting entire systems, enabling continuous improvement capabilities that define software-defined vehicles (Mobility, B.).

Virtualization and Hypervisor Technology

Virtualization technology stands central to SDV architecture. Hypervisors install virtual execution, further permitting a variety of operating systems and applications to be executed on shared hardware without imposing any serious boundary between the safety-critical and non-critical applications. This approach marks a significant evolution from traditional models, where each function required dedicated hardware.

The virtualization layer enables hardware resource consolidation, substantially reducing physical computing units while improving resource utilization. Through strong isolation between execution environments, hypervisors protect safety-critical functions from interference by entertainment applications, maintaining safety integrity despite increasing system complexity. This isolation simplifies certification processes by limiting safety analyses to clearly defined execution environments rather than requiring certification across entire systems. Additionally, compartmentalization enhances cybersecurity by containing potential security breaches within specific execution environments (Porter, D. and Muendler, Y. 2025).

Heterogeneous Computing

Modern SDV platforms leverage diverse computing architectures, combining different processor types to optimize performance and efficiency. This approach recognizes that different automotive functions have distinct computational requirements best served by specialized processing technologies.

General-purpose Central Processing Units handle supervisory functions and sequential processing tasks, while Graphics Processing Units accelerate parallel processing workloads such as computer vision algorithms essential for advanced driver assistance. Digital Signal Processors efficiently handle sensor data from cameras, radars, and other sensors throughout modern vehicles. The growing importance of artificial intelligence has driven the integration of specialized Neural Processing Units, optimizing AI workloads for tasks like object recognition, delivering substantial performance improvements compared to general-purpose processors (Porter, D. and Muendler, Y. 2025).

It is a combo method of computing that allows vehicle systems to effectively satisfy a large set of various computational needs and effectively controls power consumption, which is vital when considering both electric and standard vehicles. Heterogeneity of computing resources necessitates complex system-on-chip integrations and programs in order to efficiently divide and assign work to suitable processing components, with the aim of optimal performance and efficiency.

Publisher: SARC Publisher

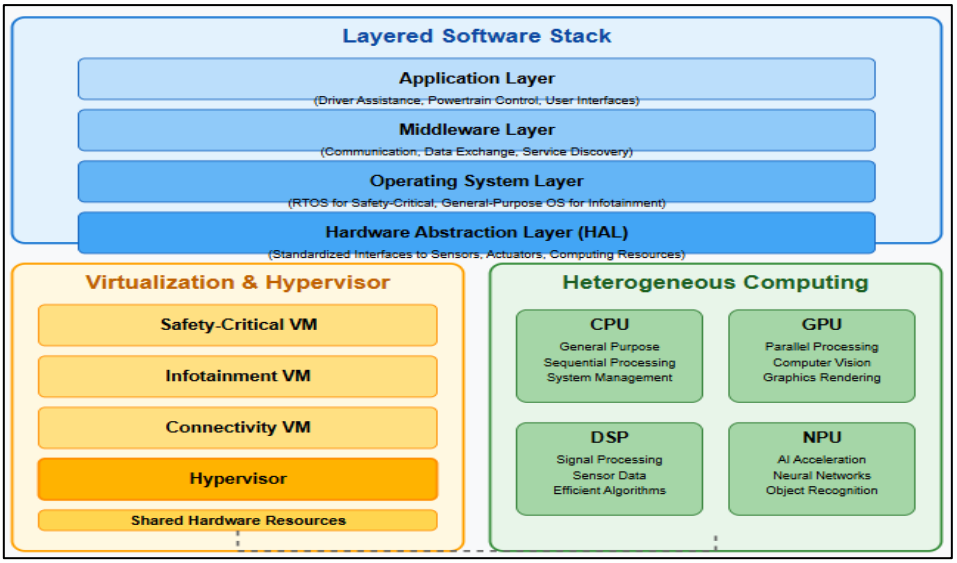

Fig 2: Core Components of Software-Defined Vehicle Architecture (Mobility, B.; Porter, D. and Muendler, Y. 2025)

ENABLING TECHNOLOGIES

Service-Oriented Architecture (SOA)

Service-oriented architecture has been adopted in cars today, and the functions are provided as distinct services speaking common interfaces. This building-block approach lets developers fix or upgrade individual features without messing up the whole system.

This service-focused strategy breaks sharply from old-school automotive software design that relied on tightly linked functions. SOA splits vehicle functions into standalone services with clear boundaries, data formats, and performance expectations. Each service handles specific jobs—like cruise control, battery management, and climate settings—while staying independent from other systems. This approach brings remarkable flexibility, allowing carmakers to improve specific features without touching everything else. Standard interfaces make it easier to mix components from different suppliers while reducing the tangled connections that made automotive software so painful to maintain. SOA also makes better use of computing power by running services on different processors based on their needs and importance. This flexibility becomes crucial as vehicles move away from scattered controllers toward central computing platforms with different performance levels and safety needs (BlackBerry QNX.).

Over-the-Air (OTA) Update Infrastructure

Perhaps the most game-changing aspect of smart vehicles is their ability to receive wireless software updates. OTA systems need sophisticated tech stretching from cloud servers all the way to in-vehicle update mechanisms.

Making OTA work properly demands many specialized parts working together perfectly. Secure data channels with heavy-duty encryption protect against hackers. Backup storage creates safety nets for updates, keeping trusted software versions that can be restored if something goes wrong. Smart rollback features let vehicles automatically return to previous versions if problems pop up after updates. Digital certificates verify software packages before installation, blocking unauthorized code. Clever delta updates transmit only what's changed between versions instead of entire packages, saving data while speeding up the process. These technical capabilities need rigorous testing to ensure updates maintain vehicle safety and reliability. Market research shows OTA capabilities becoming increasingly critical as vehicles incorporate more software features, with the automotive OTA market growing rapidly as manufacturers look for ways to enhance capabilities and fix issues remotely throughout product lifespans (Bertoncello, M. *et al*., 2021).

Middleware Frameworks

Automotive middleware frameworks like AUTOSAR Adaptive Platform and ROS provide standard ways for software components to communicate. These frameworks help components from different suppliers work together while reducing integration nightmares.

The evolution of automotive middleware shows the industry moving toward standardized, flexible

Publisher: SARC Publisher

software structures. Traditional AUTOSAR Classic Platform standardized microcontroller-based ECUs but mostly supported fixed configurations with limited flexibility at runtime. The newer AUTOSAR Adaptive Platform extends these standards to support high-performance computing with dynamic configurations, service-based communication, and runtime adaptability needed in modern smart vehicles. The development assists developers in the construction of advanced distributed systems and remains compatible with the current automotive software. Simplifier frameworks such as ROS, initially developed to serve the robotics sector, are also seeing increasing adoption in the automotive industry, with self-driving features taking advantage of its diverse ecosystem of perception, planning, and control modules. These middleware layers remove the complexities of the process-to-process communication, resource management, and backplanes to hardware, allowing the developer to concentrate on vehicle capabilities instead of backplashing. Moreover, standardization can be of particular value in dealing with the complexity of present-day automotive software, increasingly involving various suppliers developing various components that should function together without flaw (BlackBerry QNX.).

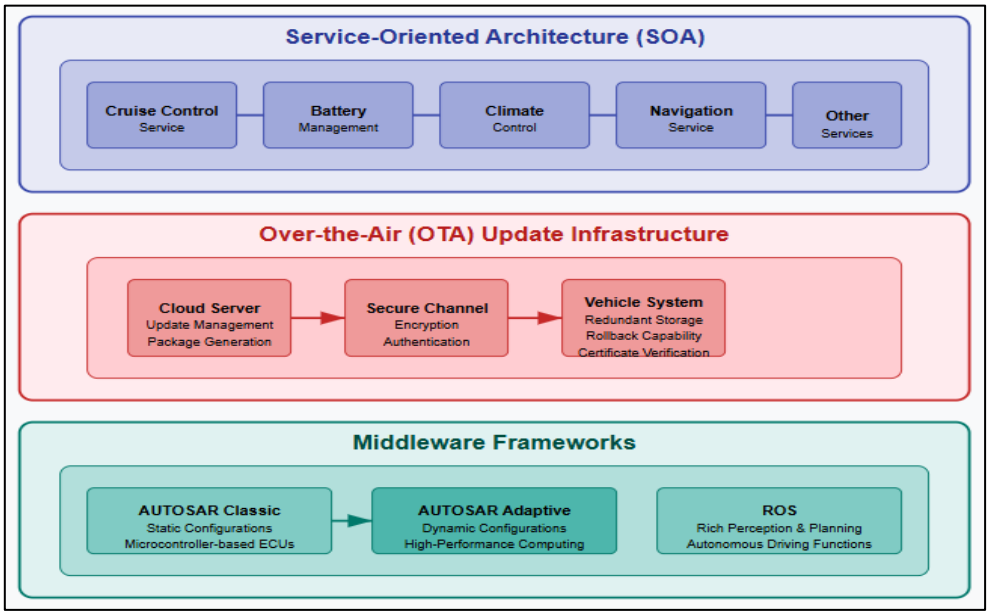

Fig 3: Enabling Technologies for Software-Defined Vehicles (BlackBerry QNX.; Bertoncello, M. *et al*., 2021)

REAL-WORLD APPLICATIONS AND BENEFITS

Accelerated Innovation Cycles

Software-defined cars are extremely fast in terms of innovation due to the modular architecture. The software updates allow Carmakers to add newer features without having to wait to introduce new hardware. This capability has shrunk traditional automotive development timelines from years to months or even weeks for certain features.

Software-defined approaches radically change automotive innovation by separating software development from hardware production cycles. Traditional vehicle development required integrating new features into next year's model, with innovation cycles measured in years. Software-defined architecture allows constant development in that features may be in development, being tested, and released over-the-air to the existing vehicle. This ability enables manufacturers to be responsive to market needs and rivalry situations, besides adding constant value to proprietors. According to industry studies, this shift leads to significant alterations of product development strategies, as greater focus is placed on agile frameworks and practices of continuous integration developed in the technological sector. The market is moving to software update models that operate on common schedules, and even presently, manufacturers are pushing periodic updates to vehicles every month or every quarter. Such a fast rate assists firms in adapting to the trends and changes in the market, the evolution in the industry, and customer response in such a way that the whole process of automotive product planning and development is altered radically (PwC. 2023).

Publisher: SARC Publisher

Cost Reduction Through Hardware Consolidation

Through the compression of several ECUs into central controllers, the manufacturers reduce the costs of using hardware and wires, which lessens the overweight of vehicles. Previously high-end cars that required more than 100 ECUs are operating using a few potent computing platforms.

The financial benefits of hardware consolidation go far beyond component cost savings. Reducing ECU count simplifies manufacturing and assembly, cutting production complexity and associated costs. Simplified wiring—achieved by consolidating functions into domain controllers connected by high-speed networks—reduces weight, improves packaging efficiency, and boosts reliability by reducing physical connection points. This consolidation improves long-term maintenance costs by reducing component variety that might need replacement over vehicle lifetimes. Beyond direct economic benefits, centralized computing enables more sophisticated software implementations, including cross-domain features that proved difficult to implement in distributed architectures. Consolidated computing resources provide extra processing power for future feature enhancements, creating more future-proof vehicle designs. This approach aligns with broader industry trends toward vehicle simplification and modularization, enabling more efficient development and manufacturing while simultaneously improving vehicle capabilities (Renault Nissan Technology & Business Centre India, 2025).

New Business Models

The software-defined approach enables entirely new business models, including feature-on-demand offerings where customers purchase or subscribe to new capabilities, data-driven services leveraging vehicle sensors and connectivity, continuous improvement of existing features throughout vehicle lifecycles, and third-party application ecosystems similar to smartphone app stores.

The ability to dynamically enable vehicle features creates significant new revenue opportunities throughout ownership periods. Manufacturers can offer feature-on-demand services where capabilities activate through software rather than requiring hardware installations, letting customers purchase or subscribe to functions based on changing needs and preferences. This approach enables more personalized experiences while creating ongoing revenue streams beyond initial vehicle sales. Market research suggests this business model transformation could increase lifetime revenue per vehicle by 15-30% compared to traditional one-time purchase models. The value of connected vehicle data leaves more room for monetization in terms of predictive maintenance services, usage-based insurance, and location-based services as well. The continuous improvement capability inherent in software-defined architecture lets manufacturers enhance existing features over time, potentially increasing satisfaction and loyalty throughout ownership experiences. Some manufacturers explore ecosystem approaches, letting third-party developers create vehicle applications and services, potentially creating new marketplace revenue streams and innovation sources beyond internal capabilities (PwC. 2023).

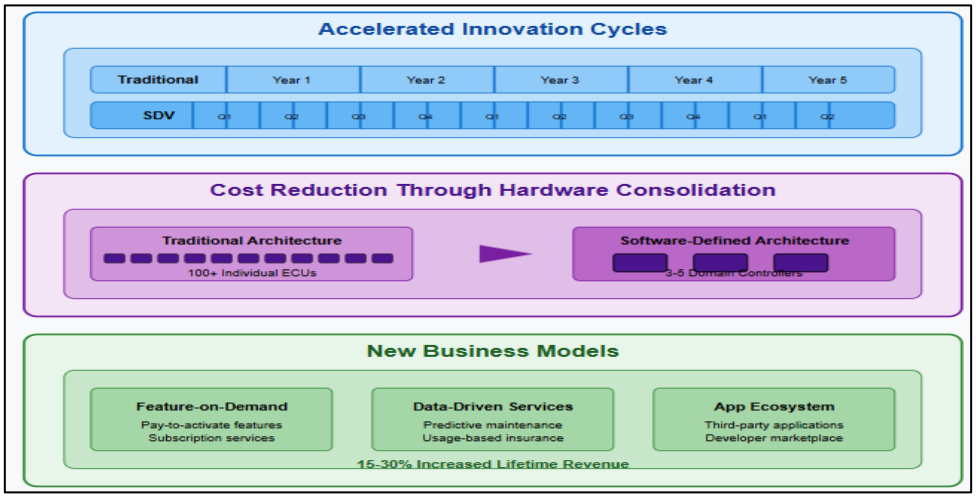

Fig 4: Real-World Applications and Benefits of Software-Defined Vehicles (PwC. 2023; Renault Nissan Technology & Business Centre India, 2025)

Publisher: SARC Publisher

CHALLENGES AND CONSIDERATIONS

Functional Safety Compliance

When a software that requires life-saving is being developed, it must be by such standards as the ISO 26262. Intelligent vehicle design will need to ensure that safety promises remain cast in stone, but enabling software in automobiles to evolve quickly reduces it to a very stiff balancing act.

The clash between rapid software updates and uncompromising safety needs creates major headaches when building software-defined cars. ISO 26262 lays out exhaustive safety rules throughout development. The standard sets Automotive Safety Integrity Levels ranging from A to D, where D means the highest safety requirements for functions that could kill or seriously injure people if they fail. Safety becomes ridiculously complex in software-defined systems where functionality varies over the lifetime of a car and where potentially completely different applications with different levels of criticality run on the same computing hardware. It is particularly difficult to keep these aspects of safety-critical and entertainment functions separate on common computational systems and requires advanced isolation methods and vigilance to ensure they do not interfere. As practice demonstrates, any safety arrangement requires serious consideration in relation not only to hardware but to software aspects as well (to which supportive systems, fault detection solutions, and graceful degradation strategies must be attributed). Testing these safety-critical systems brings extra challenges, with manufacturers using extensive testing methods, including simulations, hardware-in-the-loop testing, and real-world validation, making sure safety holds up across software changes. Safety can't be bolted on as an afterthought—it must be baked into development from day one, demanding close teamwork between systems engineers, coders, and safety experts (Horiba,).

Cybersecurity Concerns

As cars get more connected and software-driven, they face growing hacking risks. Smart vehicle designs use layered defense strategies with multiple security barriers, secure startup processes, and intrusion detection systems to fight these threats.

The extremely enlarged attack surface of networked, software-based automobiles requires the construction of end-to-end protection strategies that block all forms of attacks. The current cars have numerous points of connection, cellular networks, Wi-Fi access, Bluetooth, and special communication channels; all of them provide possible attack vectors to the attackers. According to the security studies, car hacking attempts have increased by 38 percent from 2021 to 2022, with more advanced methods of attack being directed at vehicles and the supporting infrastructure. Keyless entry systems, apps installed in smartphones that connect them with vehicles, and entertainment systems are all common weak spots, demonstrating a wide variety of threats modern vehicles have to contend with. Vulnerabilities in third-party components create special problems, with supply chain security becoming increasingly crucial for manufacturers building software-defined vehicles. Top security frameworks use defense-in-depth approaches with multiple protection layers, including specialized hardware security chips handling encryption, secure communication channels protecting data in transit, and application isolation limiting the damage from compromised components. These technical safeguards need supporting organizational measures, including threat tracking programs, vulnerability management systems, and incident response teams tackling emerging threats throughout a vehicle's life. The auto industry increasingly adopts the security-by-design principle, embedding security from the earliest development stages, recognizing that trying to add security to existing systems is far harder than building it in from scratch (Upstream Security, 2023).

Development Complexity

Transition to software-defined vehicles requires new developmental procedures, tools, and abilities. Agile development, continuous integration/ continuous deployment, and DevOps practices are some of the procedures that Carmakers are borrowing from the software world.

The people and process challenges of software-defined vehicle development often outweigh the technical hurdles. Traditional automotive development used V-model processes with sequential phases and heavy up-front planning, while software-defined approaches need more iterative and collaborative methods. This shift represents a massive cultural and operational change for organizations with decades of hardware-focused development history. Blending diverse technologies, including embedded systems, cloud computing, artificial intelligence, and

Publisher: SARC Publisher

connectivity, demands development teams with mixed expertise rarely found in traditional auto engineering groups. Testing approaches must evolve to handle software-defined system complexity, relying more on automated testing frameworks, simulation environments, and continuous validation methods, ensuring quality across frequent updates. Managing configurations becomes especially challenging, with complex relationships between software components, hardware platforms, and vehicle variants requiring sophisticated tools to maintain consistency across development. Leading automakers are building specialized development environments specifically for automotive software, with features supporting requirements tracking, model-based code generation, automated testing, and deployment management tailored to unique vehicle needs. These tool investments typically come with major organizational changes, including specialized software centers, new career paths valuing software expertise, and fresh collaboration models enabling better coordination between hardware and software teams (Horiba,).

FUTURE OUTLOOK

The smart vehicle architecture continues to evolve, and the latest technologies put it further and introduce new possibilities of automotive innovation. Since what cars can do is more determined by software, architectures are evolving to deal with the requirements of doing more complex computing, increasing connectivity, and more intelligent features.

Edge-Cloud Hybrid Computing splits workloads smartly between onboard systems and cloud servers. Time-critical functions stay local for instant response, while data-hungry applications tap cloud resources. This approach enables sophisticated features like advanced navigation with live traffic updates, maintenance predictions based on fleet-wide data analysis, and self-improving driver assistance features. Top manufacturers are creating seamless vehicle-cloud systems that work across computing environments while connecting with broader transportation networks (Kulkarni, A.).

AI-Optimized Hardware is a dedicated processing muscle to enabling automotive applications such as vision systems, sensor fusion, and self-driving systems. The tailored silicon chips become much faster and efficient in terms of energy consumption than general processors and pass rigorous automotive reliability tests. Building AI acceleration directly into sensor processing chains creates more efficient perception systems with quicker response and lower power needs. The push for automotive-specific AI chips reflects growing recognition that generic computing platforms simply can't deliver the right mix of performance, efficiency, and cost needed for advanced vehicle features (Kochhar, N. 2024).

High-Performance Computing brings data-center-level processing power to vehicles. This computational muscle enables sophisticated functions, including advanced self-driving systems, augmented reality displays, and predictive vehicle controls. Consolidating onto powerful computing platforms creates opportunities for cross-system optimization and seamless user experiences impossible with traditional scattered computing approaches (Kulkarni, A.).

Domain-specific programming languages tackle growing software complexity while boosting development efficiency. These specialized languages embed industry knowledge directly into their design, allowing clearer expression of automotive algorithms while automatically preventing certain types of errors through built-in safeguards. These languages prove especially valuable for safety-critical applications, providing stronger performance guarantees while making verification easier (Kochhar, N. 2024).

CONCLUSION

Software-Defined Vehicle architecture represents a fundamental reimagining of automotive systems engineering that transcends traditional hardware-centric approaches. By abstracting hardware complexity behind flexible software interfaces, implementing hypervisor-based isolation for mixed-criticality workloads, and enabling continuous improvement through over-the-air updates, SDVs offer unprecedented flexibility and capability throughout the vehicle lifecycle. This architectural transformation is enabling rapid advancement across driver assistance systems, autonomous driving features, and next-generation infotainment while creating new business models that extend beyond the initial vehicle sale. Despite challenges in safety compliance, cybersecurity, and development complexity, the industry continues to evolve toward more sophisticated implementations that leverage edge-cloud computing, specialized AI hardware, and domain-specific programming approaches. As vehicles increasingly become intelligent, connected

Publisher: SARC Publisher

platforms, understanding the foundational architecture of Software-Defined Vehicles becomes essential for stakeholders across the automotive ecosystem. The transformation is well underway, with software capabilities now serving as the primary differentiator in modern vehicles—truly computers on wheels, engineered for the road ahead.

REFERENCES

1. Rinftech, "Software Defined Vehicles (SDVs): Evolution, Challenges and Future Outlook." https://www.rinf.tech/software-defined-vehicles-sdvs-evolution-challenges-and-future-outlook/
2. Charette, R. N. "This car runs on code." *IEEE spectrum* 46.3 (2009): 3.
3. Burkacky, O., Deichmann, J., Doll, G., & Knochenhauer, C. "Rethinking car software and electronics architecture." *McKinsey & Company* 11 (2018).
4. Kochhar, N. "Software-defined vehicle enables future of mobility." *Siemens Thought Leadership Blog*, (2024). https://blogs.sw.siemens.com/thought-leadership/software-defined-vehicle-enables-future-of-mobility/
5. Mobility, B. "Vehicle-centralized, zone-oriented E/E architecture with vehicle computers." https://www.bosch-mobility.com/en/mobility-topics/ee-architecture/
6. Porter, D. and Muendler, Y. "Software-Defined Vehicles Shift the Future of Automotive Electronics Into Gear." *Texas Instruments White Paper*, (2025). https://www.ti.com/lit/wp/slyy236b/slyy236b.pdf
7. BlackBerry QNX, "Software-Defined Vehicles." https://blackberry.qnx.com/en/ultimate-guides/software-defined-vehicle
8. Bertoncello, M. *et al.* "Unlocking the full life-cycle value from connected-car data," *McKinsey Center for Future Mobility*, (2021). https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/unlocking-the-full-life-cycle-value-from-connected-car-data
9. PwC, "Digital Auto Report," (2023). https://www.pwccn.com/en/automotive/digital-automotive-report-2023-en-1.pdf
10. Renault Nissan Technology & Business Centre India, "Accelerating the Future of Mobility with Software-Defined Vehicles," *LinkedIn*, (2025). https://www.linkedin.com/pulse/accelerating-future-mobility-software-defined-vehicles-fwuwe
11. Horiba, "Software Defined Vehicles (SDVs)." https://www.horiba.com/ind/mobility/applications/engineering-consultancy-and-test/functional-safety/software-defined-vehicles-sdvs/
12. Upstream Security, "2023 Global Automotive Cybersecurity Report." https://www.osintme.com/wp-content/uploads/2023/03/2023_GLOBAL_AUTOMOTIVE_CYBERSECURITY_REPORT.pdf
13. Kulkarni, A. "Reimagining mobility with SDV architecture," *Tata Consultancy Services*. https://www.tcs.com/insights/blogs/future-software-defined-vehicles
14. Kochhar, N. "Software defined vehicle enables future of mobility," *Siemens*, (2024). https://blogs.sw.siemens.com/thought-leadership/software-defined-vehicle-enables-future-of-mobility/

Source of support: Nil; **Conflict of interest:** Nil.

Cite this article as:
Atmakuri, N. B. "Demystifying Software-Defined Vehicle Architecture for Modern Automotive Systems" *Sarcouncil Journal of Engineering and Computer Sciences* 4.8 (2025): pp 67-75.

Publisher: SARC Publisher